199-1299

THEME-BASED SYSTEM AND METHOD FOR CLASSIFYING DOCUMENTS

Technical Field

The present invention relates generally to the field of document classification, and more particularly, to a method and system for classifying documents automatically using themes of a predetermined classification.

Background

Companies, and in particular, companies in technical fields classify various information to various classifications for the company archives and for other purposes. One such other purpose is for patent searching. Companies may obtain various patents to store into a company archive. Commonly this is done using a group of trained searchers that read and classify the patents according to a pre-identified classification system. One problem with such a system is that the searchers must be familiar with the classification system and the underlying technology to properly classify the document. This is a very labor intensive and costly process because a substantial amount of time is required to classify the documents.

As technology changes, it may be desirable from time to time to change classification systems or add subclasses within various classifications. To accomplish this in a manual fashion would require

searchers a substantial amount of time to re-read the patents or other documents in a class and classify them into a new class or a subclass.

It would therefore be desirable to provide a classification system capable of automatically determining the classifications of documents and capable of reclassifying documents when reclassification or division of classes is desired.

Summary of the Invention

It is therefore one object of the invention to provide a classification system capable of automatically classifying documents.

In one aspect of the invention a method for classifying documents comprises the steps of:

defining a plurality of classes;

identifying source documents of each of said plurality of classes;

generating a classification theme for each of said classes;

entering an unclassified document into the system;

generating an unclassified document theme corresponding to said source documents; and

classifying the document into one of said plurality of classes when the unclassified document theme is substantially similar to the class theme score.

In a further aspect of the invention, a classification system including a controller, a

document storage memory, and a document input is used to unclassify documents. The controller is programmed to generate a theme score from a plurality of source documents in a plurality of predefined source documents. A theme score is also generated for the unclassified document. The unclassified document theme score and the theme scores for the various classes are compared and the unclassified document is classified into the classification having the nearest theme score.

One advantage of the invention is that preclassified documents may be reclassified into various classes or subclasses automatically. Therefore, as technology changes, the classes and subclasses may be updated.

Other objects and features of the present invention will become apparent when viewed in light of the detailed description of the preferred embodiment when taken in conjunction with the attached drawings and appended claims.

Brief Description Of The Drawings

Figure 1 is high level block diagrammatic view of a classification system according to the present invention.

Figure 2 is a classification hierarchy illustration according to the present invention.

Figure 3 is a block diagram of a document according to the present invention.

Figure 4 is a flow chart classification process according to the present invention.

Figure 5 is a flow chart of a reclassification process according to the present invention.

Figure 6 is a feature vector of a document.

Figure 7 is a table showing examples of information gain.

Figure 8 is a support vector machine in feature space.

Figure 9 is a plot of a transducive solution in feature space.

Figure 10 is a contingency table for a category.

Figure 11 is a table showing the number of documents and various categories.

Figure 12 is a plot of precision recall break-even point versus various topics.

Figure 13 is a table of correctly classified documents in various categories.

Figure 14 is a plot of precision/recall break-even point for various systems.

Figure 15 is a plot of precision/recall break-even point for various systems.

Description Of The Preferred Embodiment

In the following figures, specific examples of various uses of the present invention are illustrated. Although patent classification is a highly suitable use, other uses of the present invention will be evident to those skilled in the art.

Referring now to Figure 1, a classification system 10 has a controller 12 that is coupled to a document storage memory 14. Controller 12 is also coupled to a document input 16. Controller 12 preferably consists of a computer that is programmed to perform the theme-based classification as described below. Document storage memory 14 stores the various documents and classifications therein. Document storage memory 14 may be comprised of various types of storage including a hard disk drive or plurality of hard disk drives coupled together. Document storage memory 14 should be capable of storing a number of documents and capable of storing further documents as documents are classified. The document storage memory 14 is preferably capable of being supplemented when additional storage capacity is desired.

Input 16 may comprise various types of input such as a scanner or a direct interface to the Internet. Input 16 provides digitally readable documents to controller 12 for classification. In one embodiment, input 16 may be coupled to the Patent Office through a web browser. Issuing patents every Tuesday may be classified automatically by controller

12 and stored in document storage memory 14. Of course, various other means for coupling controller 12 to documents would be evident to those skilled in the art including different document sources. Input 16 may, for example, comprise a CD ROM having a plurality of unclassified or crudely classified documents thereon. Controller 12 may be used to classify the documents on the CD ROM and store them within document storage memory 14 in a classified manner.

Referring now to Figure 2, an unclassified document 20 is classified into a plurality of classes; class 1, class 2, and class 3. Class 1 has two subclasses; subclass 1 and subclass 2. Although only class 1 is illustrated as having subclasses, each of the various classes may have subclasses. Also, each subclass may have further subclasses. Each class has a respective theme score: theme score 1, theme score 2 and theme score 3. Each subclass also has a respective theme score: theme score A and theme score B. The theme scores identify the theme of the class and subclass. Unclassified document is also given a theme score 4 that is compared to the theme scores of the various classes and subclasses. The unclassified document is classified into the class and/or subclasses that correspond most closely with the theme score thereof.

Referring now to Figure 3, unclassified document 20 may have a variety of sections represented by reference numerals 22, 24, and 26. Carrying through with the patent theme, section 22

may correspond to the abstract, section 24 may correspond to a description, and section 26 may correspond to the claims. Other sections may also be used such as the international or U.S. patent classifications. Of course, other sections may be delineated depending on the type of document used. As will be further described below, each document area may have different weight in the classification scheme. As illustrated, abstract section has a weight 1, description section has weight 2, and claims section has weight 3. Preferably, only selected words are used in the weighting system. As will be further described below, various nouns and verbs may be given different weight than other parts of speech.

Referring now to Figure 5, a method 30 for classifying documents is described. In step 32 a number of classes for classification is established. At the same time, if desired, a number of subclasses may also be developed in step 34. In step 36, a number of source documents are identified for each of the classes and if desired subclasses. These source documents are meant to suitably represent the technology or field of the particular class.

The source document or source documents are used to develop a theme score for the class and subclass. The theme score represents a particular value for the subject matter of the class. Various known methods may be used to generate the theme value. For example, numerous algorithms for natural language searching may be used. The natural language

search terms are developed from the source documents. Natural language search as used in the present invention is used to refer to a question, sentence, sentence fragment, single word or term which describes (in natural language form) a particular topic (theme) or the definition for the classification uses to identify the documents to be classified. The natural language terms are arithmatically weighted according to the known methods of selecting the importance of the words to obtain the theme score. In step 40, an unclassified document is entered into the system. The unclassified document is preferably in digitally readable form and may comprise a word processing document, an Internet file, or other type of digitally readable file. In step 42, which is optional, sections of the unclassified document may be weighted. This weighting will establish an importance level for various sections with respect to developing the theme score for the unclassified document. In step 44, the theme score for the unclassified document is developed. The theme score is developed in a corresponding manner to the theme score for the classification. In step 46, the theme score of the unclassified document is compared with the various classes. The unclassified document is thus categorized into the classification having the closest theme score. If the classification has subclasses, the theme score of the subclasses is compared with the theme score of the unclassified document. Once a classification and subclassification have been determined, the documents

are stored along with its classification and subclassification into the document storage memory 14.

To enhance the integrity of the system, a review of misclassified documents may also be performed. When searchers or other users of the information in the system find documents that have been misclassified they may be identified and provided a negative weight in the system. This negative weight will prevent like documents from being classified in the similar wrong class. The review of the documents for misclassified documents is performed in step 50.

Referring now to Figure 5, one advantage of the system is that subclasses and reclassification may be performed automatically to create a new subclass. A new subclass is defined in step 52. In step 54, selected documents from the class to be divided are selected for the subclass. In step 56 a new theme score for the new subclass represented by the selected documents is performed. Thus, the theme score of the unclassified document is used for comparison with the theme score for the new subclass. The other documents in the class may be re-evaluated to determined if they should be included in the new subclass.

In a similar manner, if reclassification is performed source documents are obtained for the new classes. Each of the documents in the system may

then be re-evaluated to determine the particular new class into which they should be categorized.

EXAMPLE

An example of the present invention is set forth below:

Documents, which typically are strings of characters, have to be transformed into a representation suitable for the learning algorithms and the classification tasks. The methods that we have examined are based on the vector space method in which each document is represented as a vector of words or attributes. Each distinct word corresponds to an element of the vector whose numerical value is the number of occurrences of the word in the document. Figure 6 shows an example feature vector of a particular document. Notice that word order is lost in this representation, only word frequency is retained. Empirical research has demonstrated that classification based on statistical word counts can be quite accurate, and it has the advantage over semantic methods of not being domain specific.

Words with very low or very high frequency of occurrence in the document set, as well as a list of non-informative "stop words" are not included in the document vectors. A typical stop word list contains about 300 or 400 words including prepositions, articles, pronouns and conjunctions like "the" and "of". To improve recall and further reduce the document vector length, word stems are also used. The word stem is derived from the occurrence form of a word by removing case and

stripping suffix information. For example "compute", "computes" and "computing" are all mapped to the same stem "compute".

While *term frequency*, *TF*, gives a statistical measure to how topical a word may be in a given document, to better represent the data it has been shown that scaling the feature vector with *inverse document frequency IDF* (i.e. using TFIDF weighting) leads to an improved performance. We represent a document vector entry by f_{ij} for the term frequency of word w_i in the document d_j. $IDF(w_i)$ is defined as:

$$IDF(w_i) = log\left(\frac{N}{n_i}\right)+1$$

where N is the total number of training documents and n_i is the number of documents that contain word w_i. Intuitively, the $IDF(w_i)$ adds more influence to the feature value for the words that appear in fewer documents. Words that are common among the training documents do not appreciably help to distinguish between the documents, and therefore have small IDF weights. In order to minimize effects of document length and enhance classification accuracy, each document vector d_i is normalized to unit length.

In text categorization the dimension of the feature space (roughly the size of the vocabulary of the document set) can be quite large. Feature selection attempts to remove less representative words from the feature space in order to improve categorization effectiveness, reduce computational complexity and avoid overfitting. Feature selection

is based on a thresholded criterion to achieve a desired degree of term elimination from the full vocabulary of a document corpus. These criteria are: document frequency, information gain, mutual information, a χ^2 statistic, and term strength. The most commonly used and often most effective method is the information gain criterion. Information gain is employed from the entropy concept in information theory. Let C be a random variable over all classes and W be a random variable over the absence or presence of word w in a document, where C takes on values $\{c_i\}_{i=1}^{m}$ and W takes on values $\{0,1\}$ for the word being absent or present in a document. Information gain is the difference between the entropy of the class variable, $H(C)$, and the entropy of the class variable conditioned on the absence or presence of the term, $H(C|W)$.

$$I(C;W) = H(C) - H(C|W)$$

$$= -\sum_{i=1}^{m} Pr(c_i) \log Pr(c_i)$$

$$+ \sum_{w \in \{0,1\}} Pr(w) \sum_{i=1}^{m} Pr(c_i \mid w) \log Pr(c_i \mid w) \qquad (2.1)$$

where the probabilities are calculated by sums over all documents. Pr(ci) is the number of documents with class label ci divided by the total number of documents; Pr(w) is the number of documents containing word w divided by the total number of documents; and Pr(ci|w) is the number of documents with class label ci that contain word w divided by the number of documents containing word w. Entropy measures the uncertainty of a random variable.

Information gain measures the uncertainty reduction of category prediction by partitioning the examples according to the word. So this measure shows the importance of the word to the categorization over all the categories. By convention, 0log0 = 0. To illustrate the information gain concept, consider the small example of Fig. 7. In general, there should be enough examples to show the distribution of the probabilities. Here we have four newswire documents from categories c1 = trade and c2 = grain. We picked four words "wheat", "trade" "increase" and "export" to calculate their information gains based on the given categorization of the four documents.

The probabilities needed for calculating the information gain of the words are

$Pr(c_1) = Pr(c_2) = 1/2$

$Pr(\text{"wheat"}) = Pr(\text{"trade"}) = Pr(\text{"increase"}) = 1/2$

$Pr(\text{"export"}) = 3/4,\ Pr(\text{not "export"}) = 1/4$

$Pr(\text{not "wheat"}) = Pr(\text{not "trade"}) = Pr(\text{not "increase"}) = 1/2$

$Pr(c_1 | \text{"wheat"}) = 0;\ Pr(c_2 | \text{"wheat"}) = 1$

$Pr(c_1 | \text{not "wheat"}) = 1;\ Pr(c_2 | \text{not "wheat"}) = 0$

$Pr(c_1 | \text{"trade"}) = 1;\ Pr(c_2 | \text{"trade"}) = 0$

$Pr(c_1 | \text{not "trade"}) = 0;\ Pr(c_2 | \text{not "trade"}) = 1$

$Pr(c_1 | \text{"increase"}) = 1/2;\ Pr(c_2 | \text{"increase"}) =1/2$

$Pr(c_1 | \text{not "increase"}) = 1/2;\ Pr(c_2 | \text{not "increase"}) = 1/2$

$Pr(c_1 | \text{"export"}) = 2/3;\ Pr(c_2 | \text{"export"}) = 1/3$

$Pr(c_1 | \text{not "export"}) = 0;\ Pr(c_2 | \text{not "export"}) = 1;$

By equation (2.1), the information gain of word "wheat" with respect to the categories c_1 and c_2 can be calculated as

$$I(C;\ \text{"wheat"}) = -Pr(c_1)\log Pr(c_1) - Pr(c_2)\log Pr(c_2)$$

$$+ Pr(\text{"wheat"})\sum_{i=1}^{2} Pr(c_i \mid \text{"wheat"})\log Pr(c_i \mid \text{"wheat"})$$

$$+$$

$$Pr(\text{not"wheat"})\sum_{i=1}^{2} Pr(c_i \mid \text{not"wheat"})\log Pr(c_i \mid \text{not"wheat"})$$

$$= -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} + \frac{1}{2}(0\log 0 + 1\log 1) + \frac{1}{2}(1\log 1 + 0\log 0)$$

$$= 1$$

Similarly, the information gain of other words can be calculated and we have

$I(C;\ \text{"trade"}) = 1;$

$I(C;\ \text{"increase"}) = 0;$

$I(C;\ \text{"export"}) = 0.31$

From Fig. 7 it may be observed that the presence and absense of words "wheat" and "trade" can correctly categorize the documents and this agrees with the fact that "wheat" and "trade" have high information gain. The word "increase" shows little correlation of its presence and absense to the categorization. According to the above calculation it has information gain equal to 0. So information gain is used as a measure to select terms that best represent the categorization.

Given a training document set, for each unique term we compute the information gain, and remove from the feature space those terms whose

information gain is less than some predetermined threshold.

A classifier is a map that assigns an input attribute vector, $\boldsymbol{x} = (w_1, w_2, w_3, \ldots w_n)$, to one or more target values or classes. In our investigation we compared three methods: support vector machines, k-nearest neighbor and naïve Bayes.

Classification based upon Support Vector Machines (SVM) has developed rapidly in the last several years. It was introduced by Vapnik in 1995 for solving two-class pattern recognition problems [25]. Suppose the training data is (x1, y1), (x2, y2), …, (xl, yl), where xi is the attribute vector of document i and yi is the target value of xi which is either 1 or -1 depending on whether document i is in one class or the other. SVM, operating as a two-class classifier, is to construct an optimal hyperplane $w \cdot \boldsymbol{x} + b = 0$ that separates the data points in two classes with maximum margin such that $y_i(w \cdot x_i + b) \geq 1$, i = 1,…l. (See Figure 8).

The optimal hyperplane can be defined by the vector w_0 and the constant b_0 that minimize $\frac{1}{2} \| w \|^2$ subject to constraints $y_i(w \cdot \boldsymbol{x}_i + b) \geq 1$. When the problem is not linearly separable this method can be augmented by introducing a soft margin and mapping the training data nonlinearly into a higher-dimensional feature space via a function Φ, then construct an optimal hyperplane in the feature space.

In general with threshold T the hyperplane decision function can be determined as follows:

SVM will assign to each new input vector $\boldsymbol{x}$ a target value 1 if

$$\sum_{i=1}^{l} y_i \alpha_i (\Phi^T(\boldsymbol{x})\Phi(\boldsymbol{x}_i)) + b \geq T \quad \text{or} \quad \sum_{i=1}^{l} y_i \alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b \geq T$$

and -1 otherwise where Φ is the nonlinear map from input space to the feature space and $k(\boldsymbol{x}, \boldsymbol{x}_i) = \Phi^T(x)\Phi(\boldsymbol{x}_i)$ is the kernel function, and $\alpha_1, \alpha_2, \ldots, \alpha_l$ are the weights trained through the following quadratic optimization problem:

$$\text{minimize: } w(\alpha) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

$$\text{subject to: } \sum_{i=1}^{l} y_i \alpha_i = 0$$

$$\forall i : 0 \leq \alpha_i \leq C$$

The kernel function can be the following types of functions:

linear function

polynomial $(x \cdot y + c)^d$ of degree d

radial basis function $\exp(-\| x - y \|^2 / (2\sigma^2))$

sigmoid function $\tanh(\kappa(x \cdot y) + \Theta)$

An interesting property of SVM is that the optimal hyperplane is determined only by the data points located on the margin. These data points are called support vectors. The quadratic optimization problem stated above can be solved by a quadratic programming (QP) solver. However, many QP methods can be very slow for large problems such as text categorization. Different training algorithms that

decompose the problem into a series of smaller tasks have been developed. Relatively efficient implementations of SVMs include the SVM^{light} system by Joachims and the Sequential Minimal Optimization (SMO) algorithm by Platt. In addition to regular SVMs Joachims also introduced transductive SVMs. When there is very little training data it is crucial that the method can generalize well. Transductive SVMs take into account both training data and testing data to determine the hyperplane and margin that separate them.

Notice that SVM is a two-class classifier. To extend it to multiple classes we used a one class versus all other classes scheme, training a separate SVM classifier for each class. Note that different k-class classification schemes based on two-class classifiers have been developed.

In our experiments we used SVM^{light} which includes both regular (inductive) SVMs and transductive SVMs.

The most basic instance-based method is the k-Nearest Neighbor algorithm (kNN). The idea is very simple. Given a test document, the system finds the k nearest neighbors among the training documents, and the categories associated with the k neighbors are weighted based on the distances or the similarities of the test document and the k nearest neighbors from the training set. The category or categories with weights greater than or equal to a certain threshold are assigned to the test document as its

classification. The nearest neighbors can be found by inner products, cosines or other distance metrics. In our implementation we used cosine to measure the similarity of vectors which is defined by

$$cosine(\boldsymbol{x}, \boldsymbol{x}_i) = \frac{\boldsymbol{x} \cdot \boldsymbol{x}_i}{||\boldsymbol{x}|| \cdot ||\boldsymbol{x}_i||}$$

Since both $\boldsymbol{x}$ and $\boldsymbol{x}_i$ are normalized to unit length, $cosine(\boldsymbol{x}, \boldsymbol{x}_i) = \boldsymbol{x} \cdot \boldsymbol{x}_i$. To state the method formally, let $\boldsymbol{x}$ be a test document vector. Let $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, ..., $\boldsymbol{x}_k$ be the k nearest neighbors of $\boldsymbol{x}$ in terms of the cosines between two document vectors, and let c_1, c_2, ... c_l be the categories of the k neighbors. With the threshold T the classification of $\boldsymbol{x}$ is determined by

$$\{c_j \mid \sum_{i=1}^{k} cosine(\boldsymbol{x}, \boldsymbol{x}_i) \cdot \delta(\boldsymbol{x}_i, c_j) \geq T\}$$

where $\delta(x_i, c_j) =$ 1 if $\boldsymbol{x}_i$ is in category c_j and 0 otherwise.

The approach of Naïve Bayes is to use the training data to estimate the probability of each category given the document feature values of a new instance. Bayes theorem is used to estimate the probabilities:

$$\Pr(C = c_k | \boldsymbol{x}) = \frac{\Pr(\boldsymbol{x} | C = c_k) \Pr(C = c_k)}{\Pr(\boldsymbol{x})}$$

The category with maximum probability determines the classification of the instance. The quantity $\Pr(\boldsymbol{x} | C = c_k)$ is impractical to compute without making the simplifying assumption that the features are conditionally independent in a given class C. This yields the following

$$\Pr(\boldsymbol{x}|C = c_k) = \prod_i \Pr(\boldsymbol{x}_i|C = c_k)$$

While this assumption is generally not true for word appearance in documents, research has shown that there isn't obvious improvement when word dependency is taken into account. Once the calculation is made, a threshold can be applied such that if $\Pr(C = c_k \mid \boldsymbol{x}) \geq T$, then document $\boldsymbol{x}$ is classified in the class c_k .

Given a binary classification task, documents can be correctly/incorrectly classified as being in/out of the class as shown in the contingency table Table 2. Precision and recall are two fundamental measurements used to evaluate the performance of classifiers. Precision is the percentage of the documents classified by the system to the class that actually belong in that class. In other words, precision is a measure of how much junk is returned with the valuable information. Recall is the percentage of documents actually classified to the class among all the documents that belong in that class. In other words, recall is a measure of how much of the valuable information that is available is returned. Lowering the threshold used to determine whether or not a document belongs in a given class has the effect of increasing the recall, but decreasing the precision. Similarly, raising the threshold can improve precision at the expense of recall. By finding the precision/recall breakeven point, the threshold at which precision and recall are equal, both measures are given equal weight in the analysis.

When there are many categories to learn, a separate classifier is trained for each. Once precision/recall breakeven ratios are calculated for each category, they can be combined in either a micro-average or a macro-average. Let x_i / y_i be the precision/recall breakeven for category i which represents x_i documents among y_i documents in category i are classified to the category by the system. The micro-average is the average of total number of documents properly classified for all categories by the system over all the documents that are in the categories ($\sum_i x_i / \sum_i y_i$). The macro-average is just the average of the individual category ratios over all the categories, ($\frac{1}{n}\sum_i x_i / y_i$). The micro-average scores tend to be dominated by the classifier's performance on those categories that contain more documents, and the macro-average scores are influenced equally by the performance on all categories regardless of the number of documents they contain. We will demonstrate our results in terms of these measurements.

We ran experiments on three different data sets. The biggest data set is the reuters-21578 collection. This data set consists of newswire articles on a wide range of topics. Some of the topics include earnings, money/foreign currency exchange, grain, trade, etc.. There are about 120 different topics in the collection. Based on the Lewis split we extracted 10802 stories in which 7780

are in the training set and 3022 are in the testing set. The number of stories in each category varied widely. For example the "earnings" category contains 3965 documents while many other categories contain only one document. We only used the 10 most frequent categories for the experiments. The number of training and testing examples in these 10 categories are shown in Figure 11.

For each category in Figure 11 the numbers show the positive training and testing examples and the remaining extracted stories from the training and testing set are used as negative examples.

The exploration of text classification was originally motivated by automating Japanese patent reclassification with a developed scheme. Since we needed a set of properly classified technical patents with engineering terms, we chose some US patents on engineering from US PTO database. To reflect the real situation where categories can be very similar and for some categories there can be very few classified patents, we chose those patents from subclasses 31 and 32 in class 706 on neural networks in particular, with 50 patents in each class. Only title and abstract are included in the text analysis. To obtain training and testing data we split the two classes of 50 patents in two ways. One is randomly selecting 40 patents from each class as training documents and using the remaining 10 patents in each class as testing documents. The other way is randomly selecting 10 patents from each class as training documents and using the remaining 40 patents

in each class as testing documents. We did 20 random selections for each way of splitting the data, ran our experiments on them and averaged the results separately for the two splits.

On the Reuters-21578 collection, we obtained precision/recall breakeven point for the 10 most common categories using Naïve Bayes, k-Nearest Neighbor and Support Vector Machines. With such a large data set, the vocabulary size is very big. Considering the processing time and the training time we used a lower number of word stems to do the experiments. For all the methods we selected 500 words with highest information gain. For kNN we used the number of nearest neighbors equal to 50. For SVM we chose the kernel function to be radial basis function with gamma equal to 1. The breakeven point for the two methods for the 10 most frequent categories is summarized in Figure 12.

Support Vector Machines works the best on all ten categories, the micro-average of breakeven for the ten categories is 92.93% and the macro-average of breakeven is 85.8%. k-Nearest Neighbors gives good precision/recall breakeven point close to that of SVM on the two most common categories, its micro-average over the ten categories is 89.44% and the macro-average is 79.6%. Naïve Bayes gives the lowest breakeven point, its micro-average over the ten categories is 82.81 and the macro-average is 70.39%.

The numbers included in are from the precision/recall breakeven calculation. From the recall perspective we can see that for category Earn, 1075 out of 1088 documents in Earn category are correctly classified only 13 documents are misclassified. From the precision perspective it means that there are also 13 documents not in Earn category are classified to this category. We can vary the decision threshold to get higher precision at the cost of lower recall or higher recall at the cost of lower precision. Human intervention can also be involved to further improve the classification accuracy. Some confidence scores may be given to documents to aid the human confirmation.

On the two classes of neural network patents we compared breakeven points of SVM, Transductive SVM, kNN and Naïve Bayes by using all the word stems. The training and testing data of 50 documents were separated in two different ways. In the 40/10 split each class contains 40 training documents, in the 10/40 split there are 10 training documents in each class. By training SVMs using different kernel functions, we chose the kernel function of both SVM and Transductive SVM for the 40/10 split to be radial basis function with gamma equal to 2 and the kernel function of both SVM and Transductive SVM for the 10/40 split to be a polynomial of degree two. The number of nearest neighbors for kNN for both splits is chosen to be 5. The breakeven points of the three methods with two different ways of training and testing data split are shown in two separate diagrams in Figures 14 and 15.

In the 40/10 split, SVM shows better performance than transductive SVM and both are better than kNN. In the 10/40 split SVM performs the best among three methods. From the literature, transductive SVM works better than inductive SVM when there is very few training data and sufficient amount of testing data. This is verified using provided sample data. However, using 10/40 split, the transductive SVM doesn't provide superior performance as expected with very few training documents. This could be because the entire data set is too small to show its difference from SVM. Naive Bayes has the lowest accuracy.

We also examined all these methods by using 50% of vocabulary based on information gain. The result shows ~± 2.5% variation on the average breakeven. All three methods are very stable with respect to variation in feature dimensionality. With reasonable number of features selected both SVM and kNN generalize well. The results are not shown here.

Patents of subclasses A, B and D from are not evenly distributed. Subclass A contains only 8 documents and subclass D, the largest subclass, has 99 documents. Only 5 documents in subclass A are selected as positive training examples when classifiers are trained for subclass A, 88 documents from other two subclasses are used as negative training examples. Because of having very few positive training examples Naïve Bayes, kNN, SVM and Transductive SVM don't generalize well for subclass A. But the performances of different algorithms are

improved for the other two subclasses, especially subclass D.

In the two patent data sets the classes in each data set are very similar which makes it very hard to achieve high performance scores. However, if classes are distinct enough it is easy to get very high accuracy. We also tried to classify two classes of patents that we downloaded from US PTO database with one class on neural networks the other on fuel cells. There are 40 training documents and 10 test documents in each class. Naïve-bayes, kNN, inductive and transductive SVMs were applied. All methods produce 100% classification accuracy.

It was found that SVM is the most accurate classifier on all trials among all the classifiers considered. KNN also gives good classification performance. When classes are distinct, classifiers can be very accurate. When classes are similar, classifiers will generalize well only if sufficient training data is available. From literature and our experiments using software provided sample data, transductive SVM shows its advantage, when there is minimal training data but plentiful test data. However, our experiment on a small data set did not explore this regime. Also when the training data set gets very large kNN slows down significantly. In general, Transductive SVM takes much longer to train than inductive SVM, but the classification times of both inductive and transductive SVM are efficient. In addition both SVM and KNN are very stable with

different model parameters. Even with different kernel functions for SVM, the precision/recall breakeven points are very close. We also varied the number of nearest neighbors for kNN. The results show that optimal number of nearest neighbors is small relative to the size of training data set and feature length. For example, on Reuters-21578 collection 50 nearest neighbors were used for kNN where 500 features were chosen and the total number of training documents is 7780. The observation can be made that if the number of nearest neighbors for kNN is too large, small isolated categories may be difficult to distinguish from heavily populated categories. Even though Naïve Bayes is very efficient, it doesn't produce as accurate classification as SVM and kNN do. Results also indicate that classification with modestly lower or higher feature dimension does not appreciably affect the results.

From the above discussion we can conclude that SVM and kNN are efficient, robust and stable methods that give good classification performance. These methods can be used to automate the process of text document classification. For a small number of cases near the boundaries, misclassifications do occur. These situations can be handled by human intervention. It is also possible to introduce a degree of confidence measure to identify boundary cases. However, we note that the bulk of the classification task can be reliably automated,

greatly reducing the work load for the experienced individuals who now perform these tasks.

While particular embodiments of the invention have been shown and described, numerous variations alternate embodiments will occur to those skilled in the art. Accordingly, it is intended that the invention be limited only in terms of the appended claims.